Imperial College London

LSDS Large-Scale Distributed Systems Group

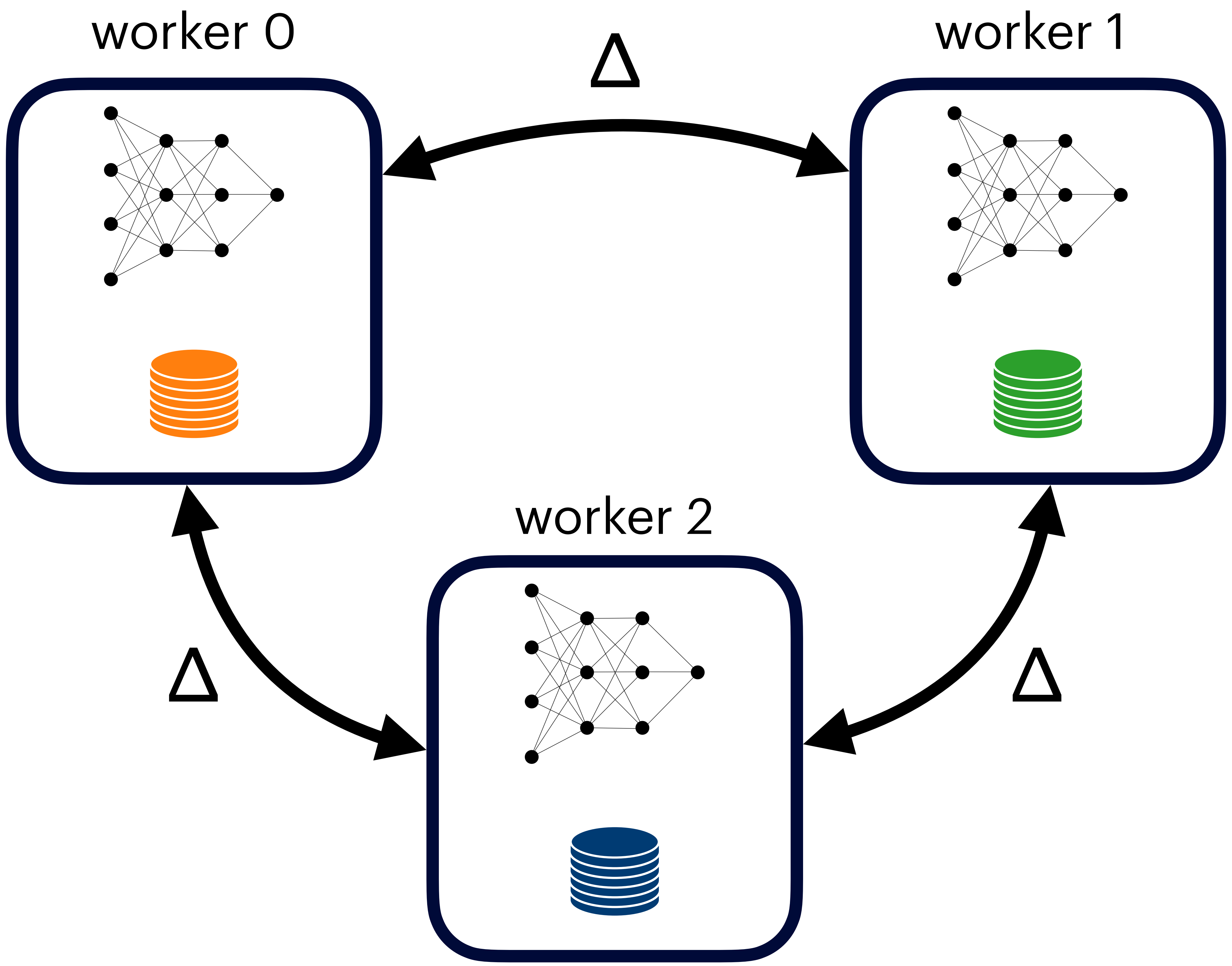
Spotnik

Designing Distributed Machine Learning for Transient Cloud Resources

Marcel Wagenländer, Luo Mai, Guo Li, Peter Pietzuch
Large-Scale Data & Systems (LSDS) Group
Imperial College London

Distributed ML

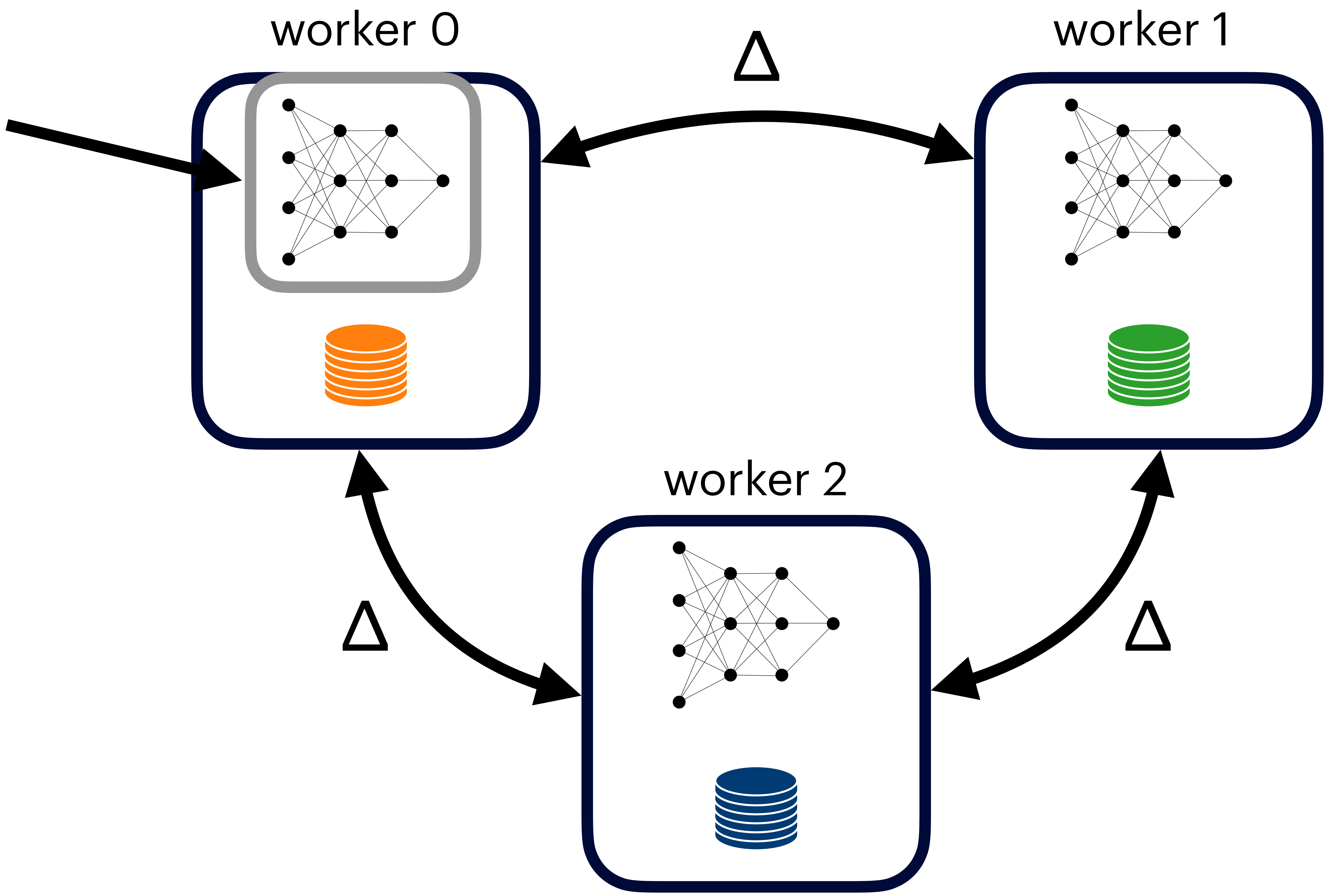
Train a machine learning model



Distributed ML

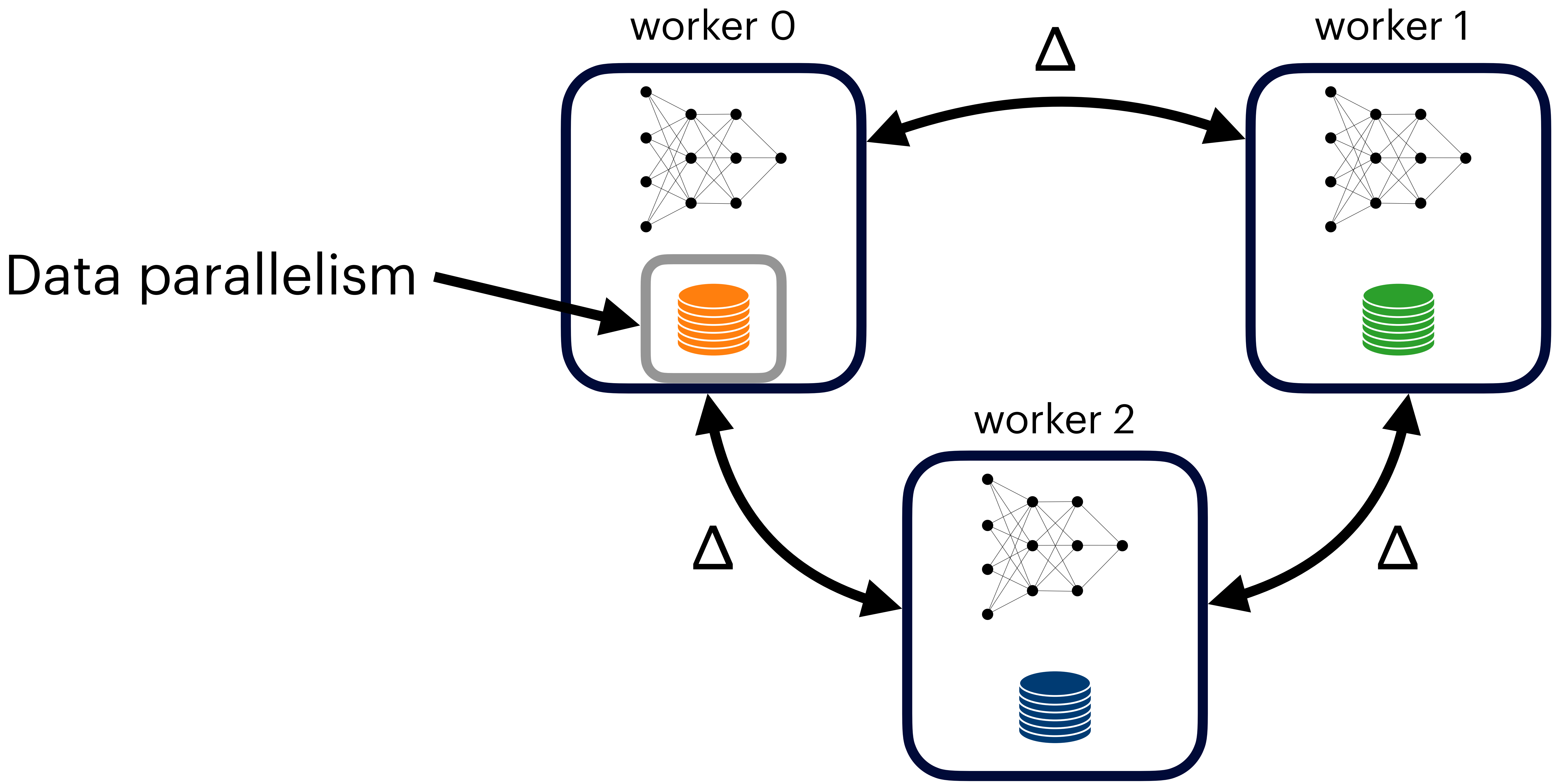
Train a machine learning model

Learn a model



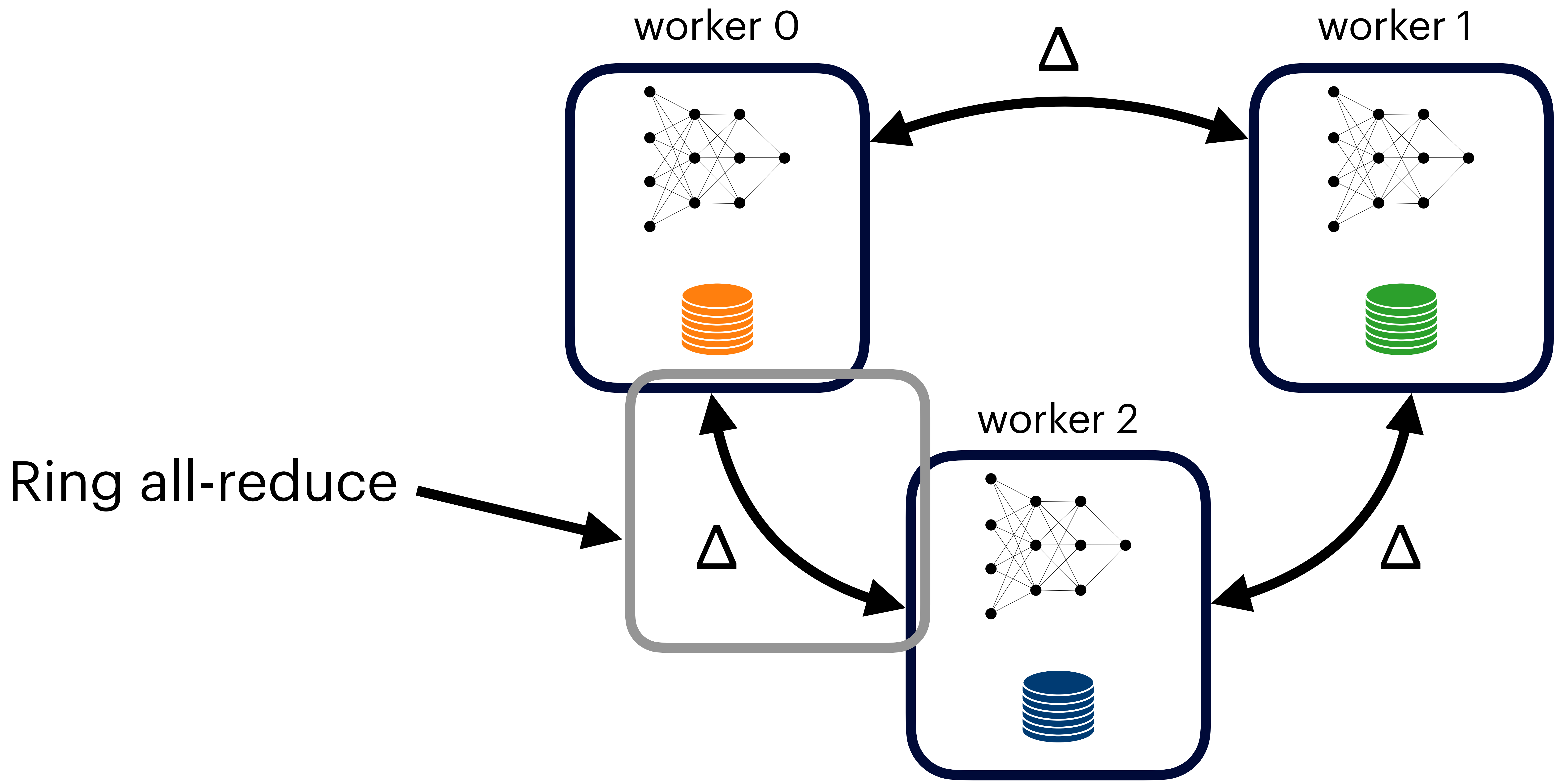
Distributed ML

Train a machine learning model



Distributed ML

Train a machine learning model



Challenges of distributed ML

- Distributed ML is **resource-hungry**
- Accelerated resources are **expensive**

Example Megatron-LM[3]

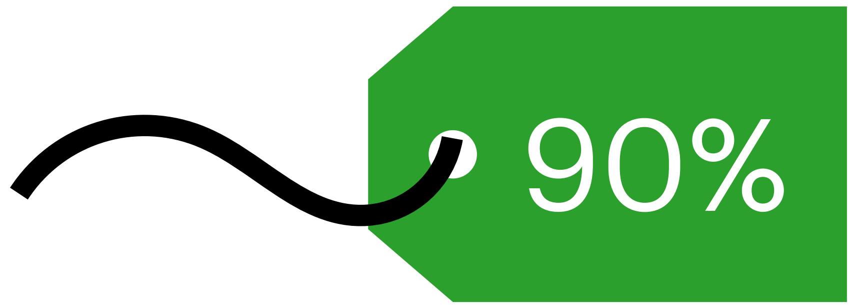
- Training of BERT-like model
- 512 V100 GPUs
- One epoch (68,507 iterations) takes 2.1 days
  - Cost on Azure: $92,613

[3]Shoeybi, Mohammad, et al. Megatron-LM: Training multi-billion parameter language models using gpu model parallelism, 2017

Transient cloud resources

- Examples: AWS Spot instances, Azure Spot VMs
- Follows the law of a free market
- Revocations
  - Notifications

- Economic incentive
  - Offers a **cost reduction** of up to 90%[1]

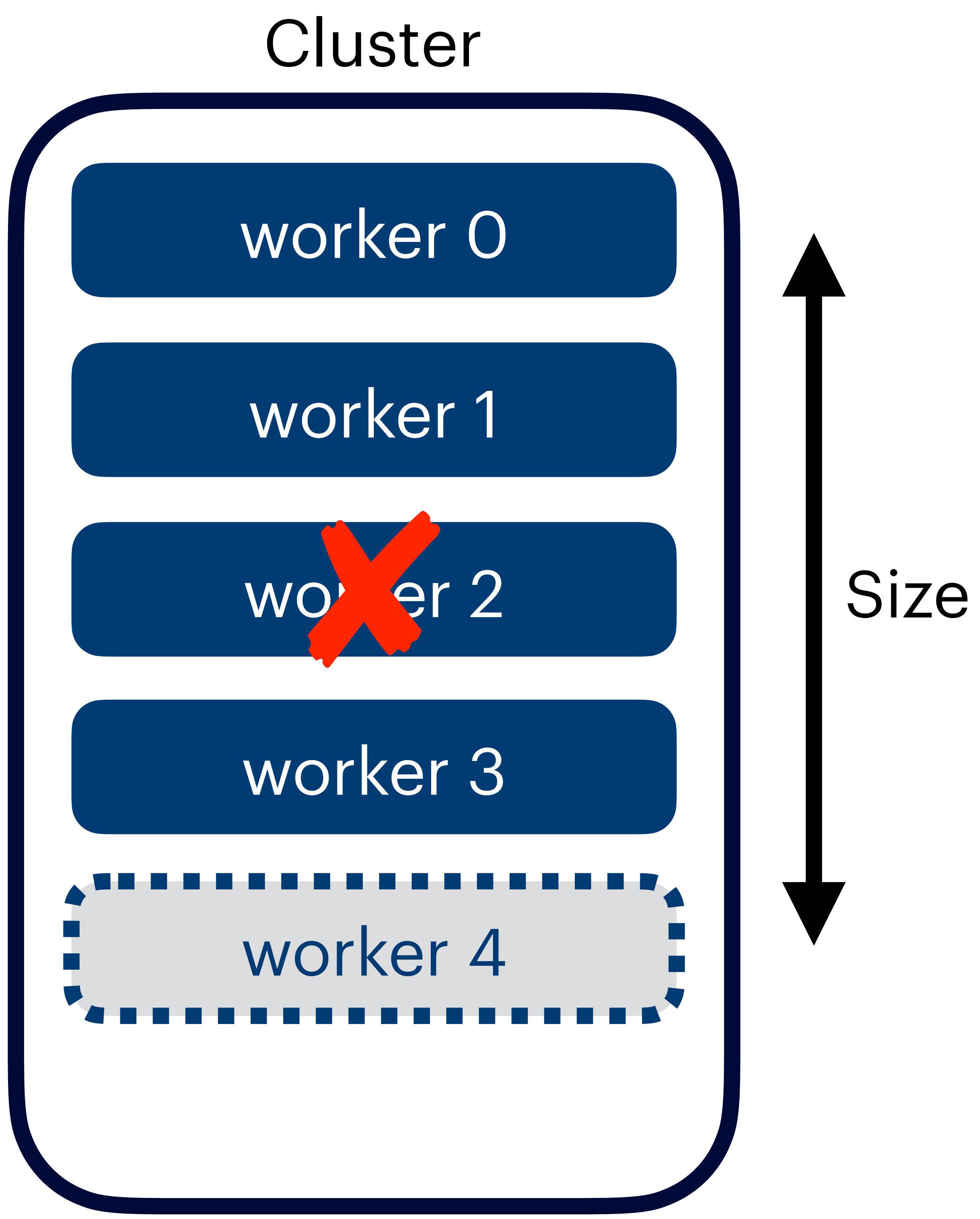


A Megatron-LM epoch would drop from $92,613 to $15,152

[1]https://azure.microsoft.com/en-us/pricing/spot/

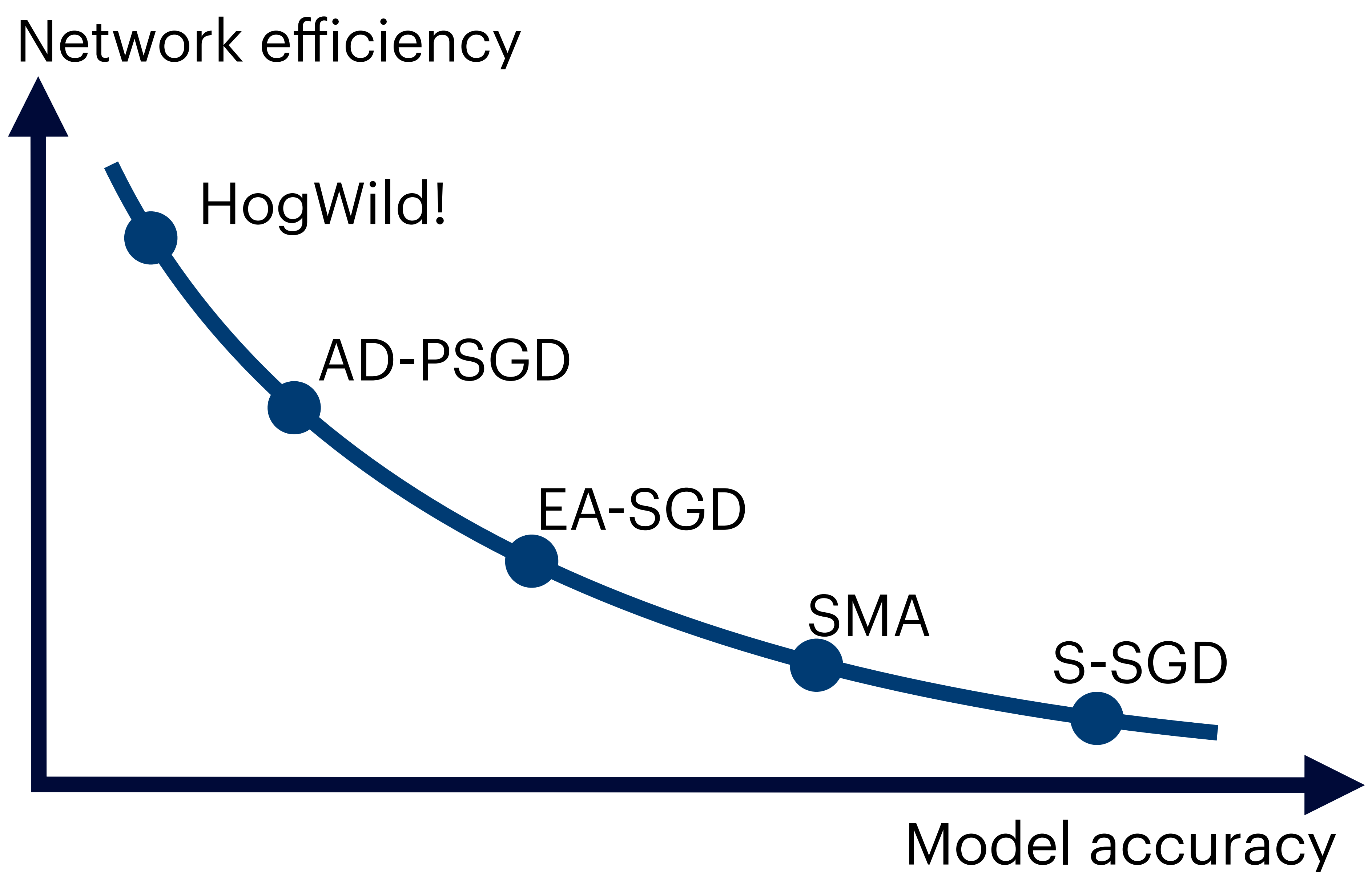
Implications of transient resources

- New workers become available or old workers get revoked
  - ➙ System must cope with disappearing resources
- Changes can happen at any time
  - ➙ System must ensure consistency of updates



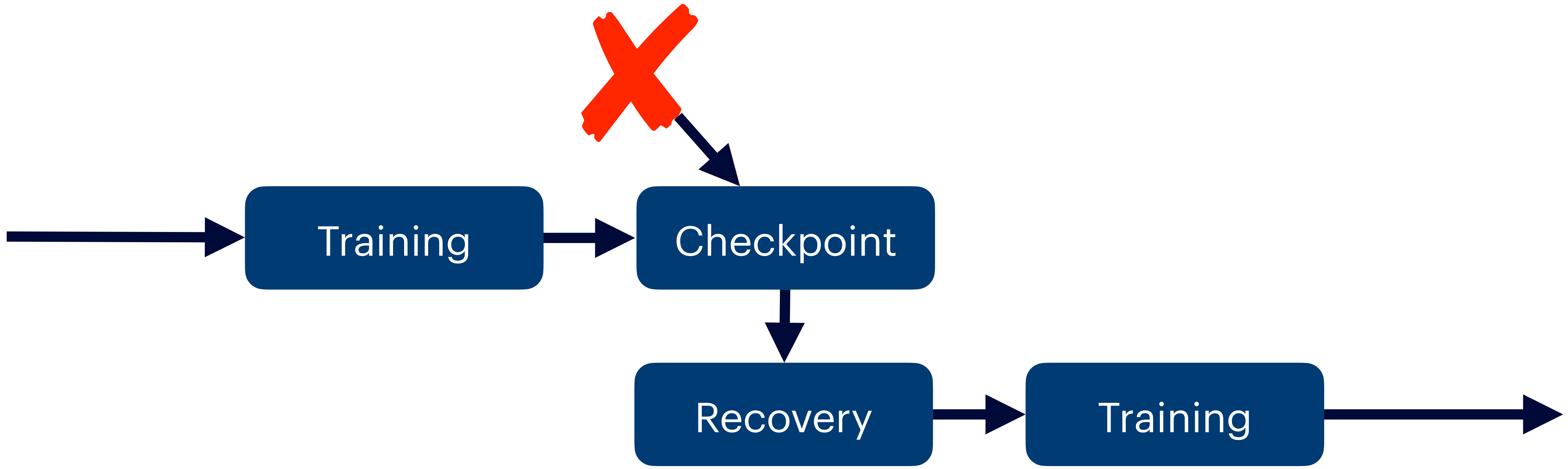
Implications of transient resources

- New workers become available or old workers get revoked
  - ➙ System must cope with disappearing resources
- Changes can happen at any time
  - ➙ System must ensure consistency of updates
- Cluster sizes are unknown beforehand
  - ➙ System must adapt to different conditions



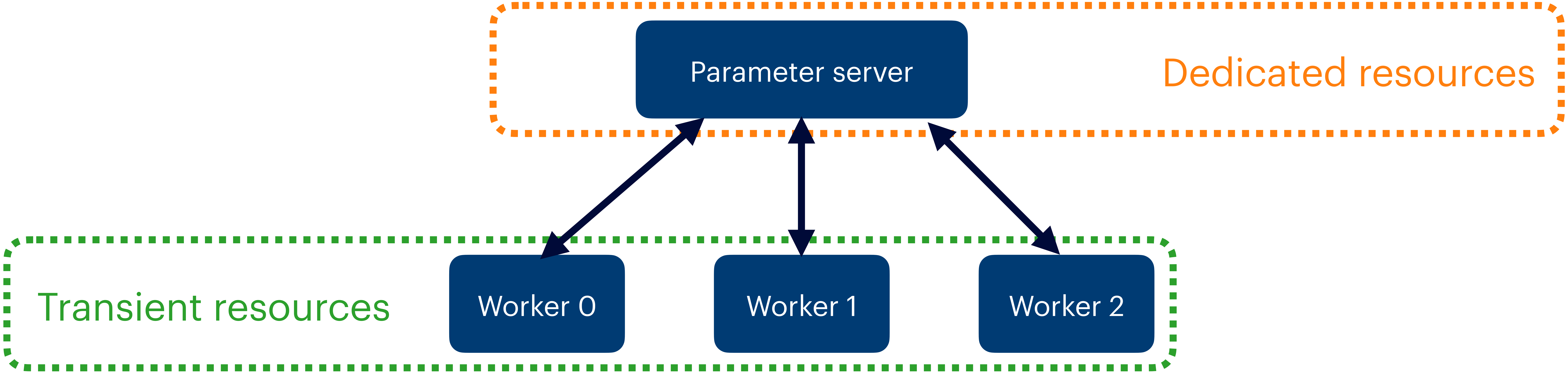
Current approach: Checkpoint & recovery

- Tensorflow and Pytorch
- Changes to the cluster are not considered
- Recovery takes about 20 seconds with ResNet50 and ImageNet



Current approaches: Hybrid

- Mix dedicated resources with transient resources
- Proteus[2]: Placement of parameter server on dedicated resources so that the training state is save



[2]Harlap et al. Proteus: agile ML elasticity through tiered reliability in dynamic re- source markets. EuroSys, 2017

Spotnik

Challenges	Solutions
Workers become available or get revoked	**Reuse communication channels** for synchronisation to repair the cluster
Changes can happen at any time	Ensure **atomic model updates** by waiting for all synchronisations to finish first
Cluster sizes are unknown beforehand	**Change the synchronisation strategy** based on the number of workers

Spotnik

Challenges	Solutions
Workers become available or get revoked	**Reuse communication channels** for synchronisation to repair the cluster
Changes can happen at any time	Ensure **atomic model updates** by waiting for all synchronisations to finish first
Cluster sizes are unknown beforehand	**Change the synchronisation strategy** based on the number of workers

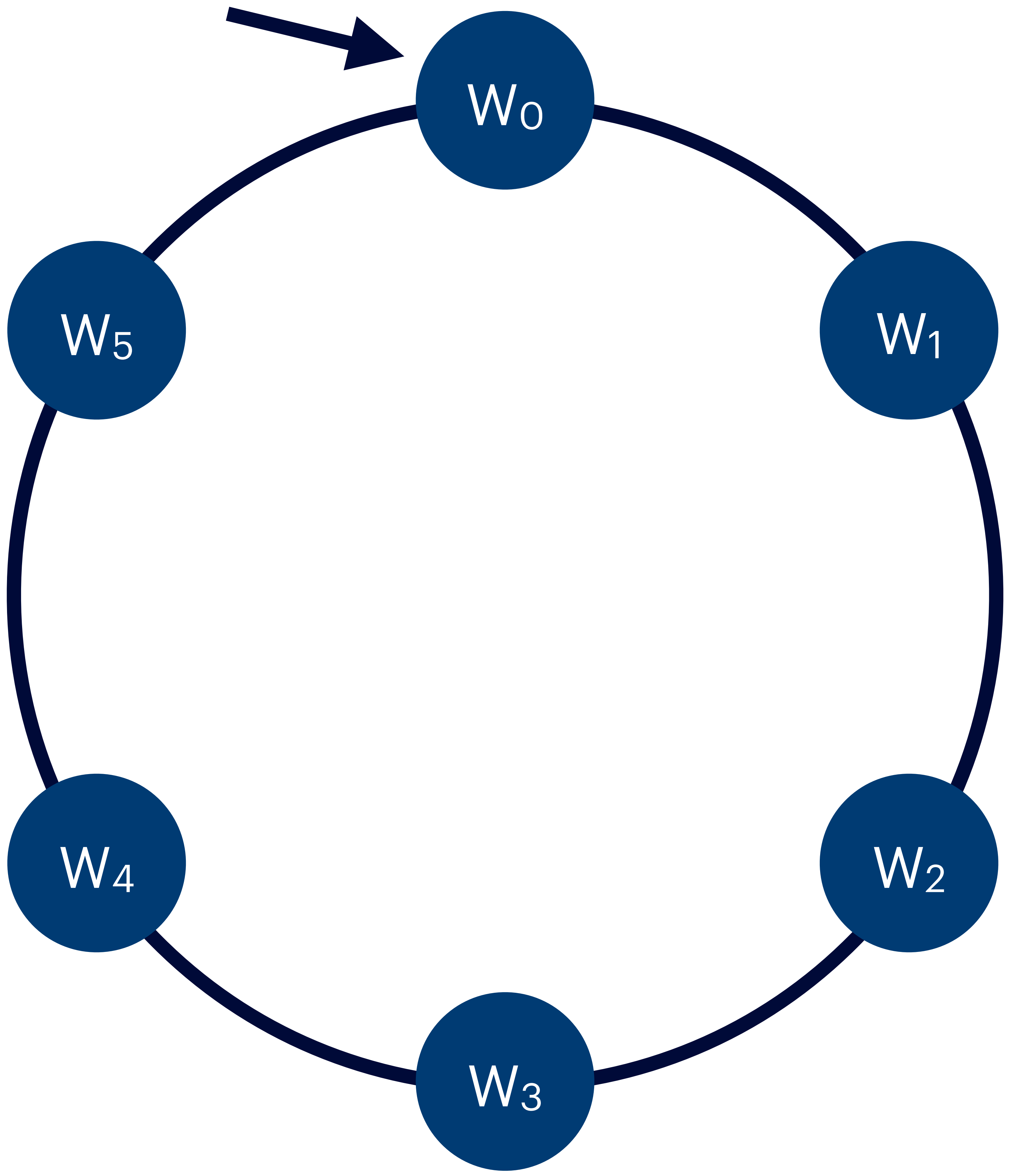
Revocation recovery algorithm

- Handle revocations within a ring topology
- Number of total messages is bounded by $O(N \cdot K)$ messages
  - K is the number of simultaneous revocations
  - N is the number of workers

  → Scale to many transient resources
- No reliance on revocation notifications

Revocation recovery algorithm

Repairing a broken all-reduce ring

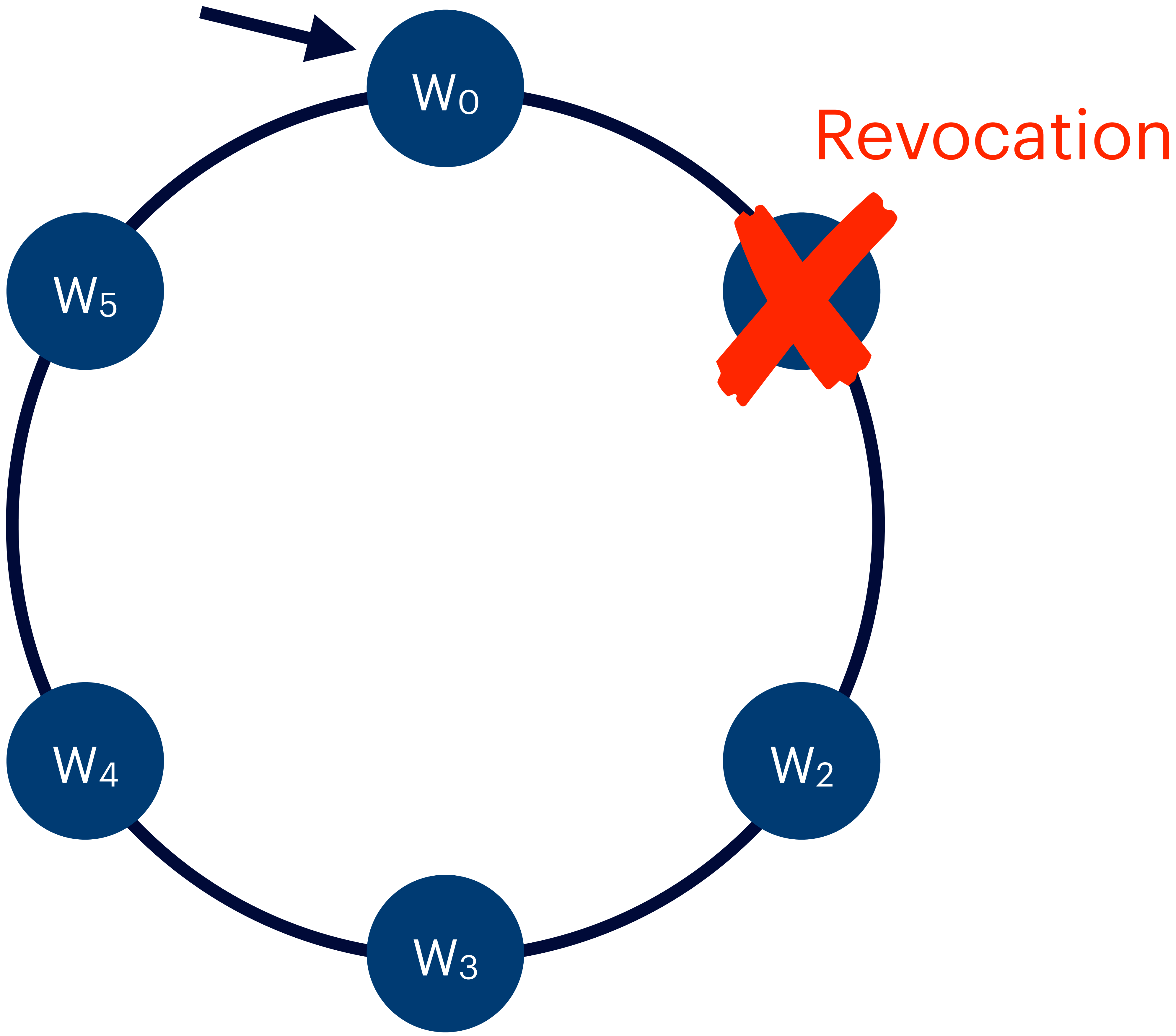


Worker	Membership
0	[0, 1, 2, 3, 4, 5]
1	[0, 1, 2, 3, 4, 5]
2	[0, 1, 2, 3, 4, 5]
3	[0, 1, 2, 3, 4, 5]
4	[0, 1, 2, 3, 4, 5]
5	[0, 1, 2, 3, 4, 5]

Revocation recovery algorithm

Repairing a broken all-reduce ring

Reconcile

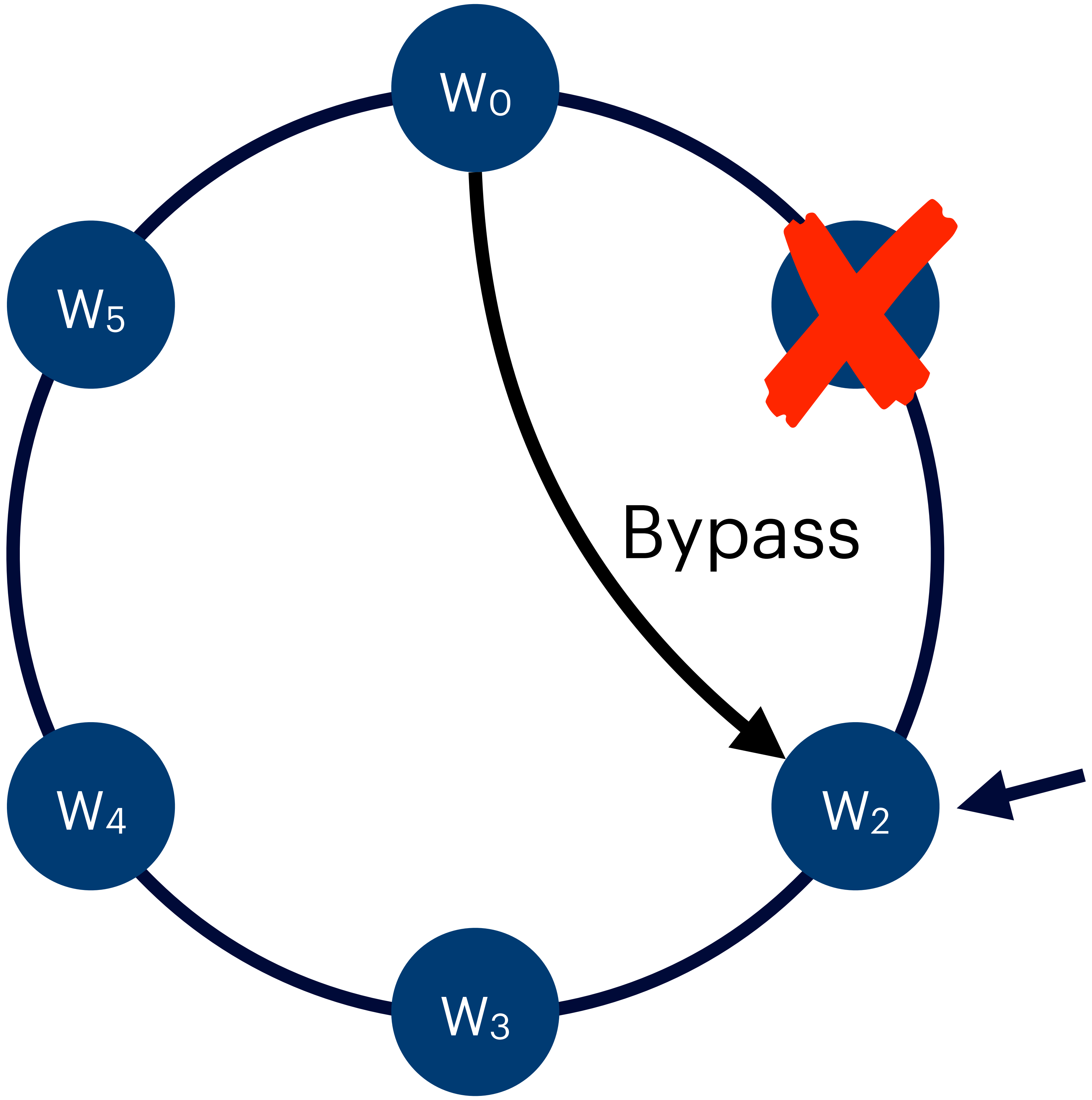


Worker	Membership
0	[0, 1, 2, 3, 4, 5]
2	[0, 1, 2, 3, 4, 5]
3	[0, 1, 2, 3, 4, 5]
4	[0, 1, 2, 3, 4, 5]
5	[0, 1, 2, 3, 4, 5]

Revocation recovery algorithm

Repairing a broken all-reduce ring

Reconcile

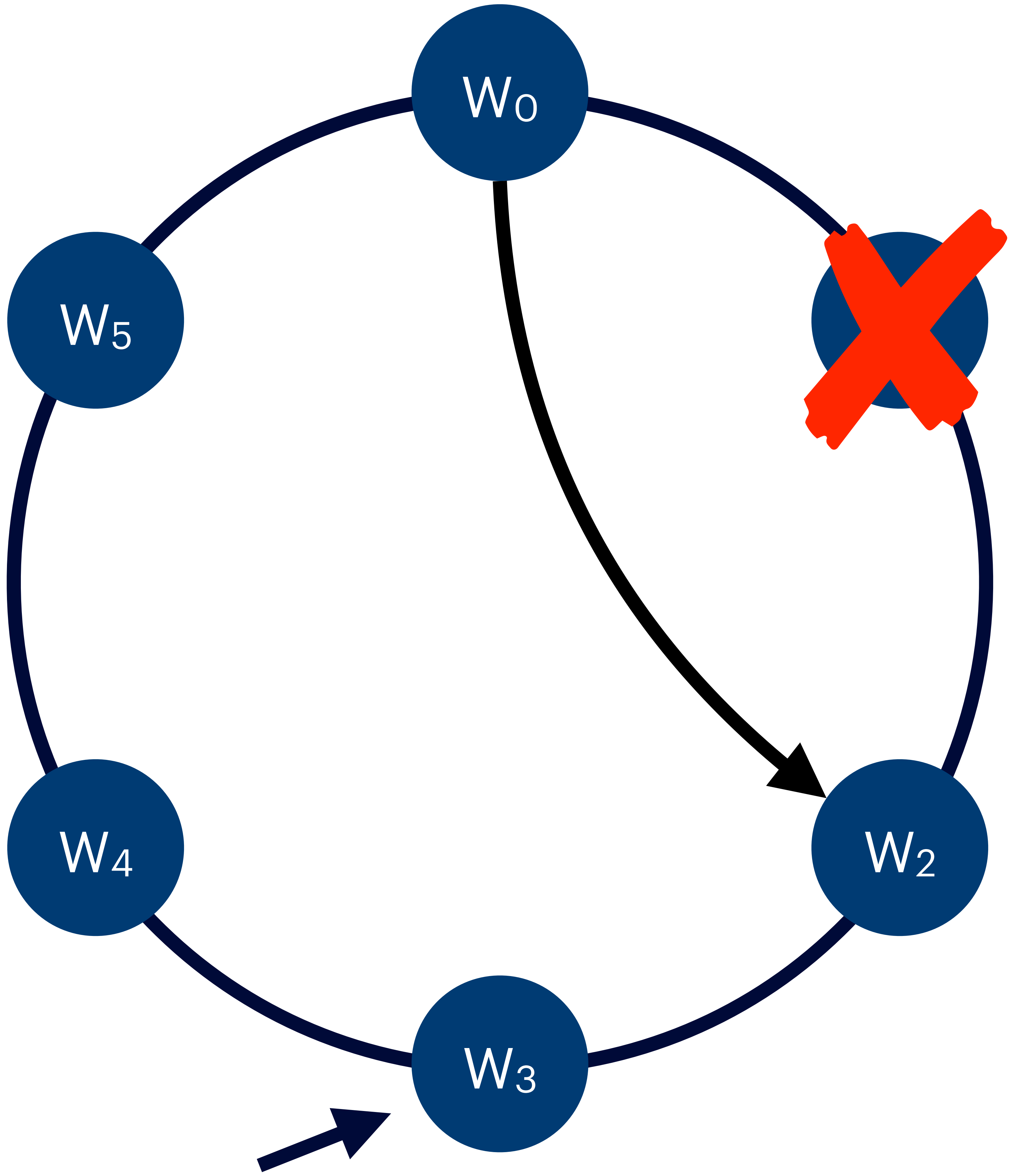


Worker	Membership
0	[0, 2, 3, 4, 5]
2	[0, 2, 3, 4, 5]
3	[0, 1, 2, 3, 4, 5]
4	[0, 1, 2, 3, 4, 5]
5	[0, 1, 2, 3, 4, 5]

Revocation recovery algorithm

Repairing a broken all-reduce ring

Reconcile

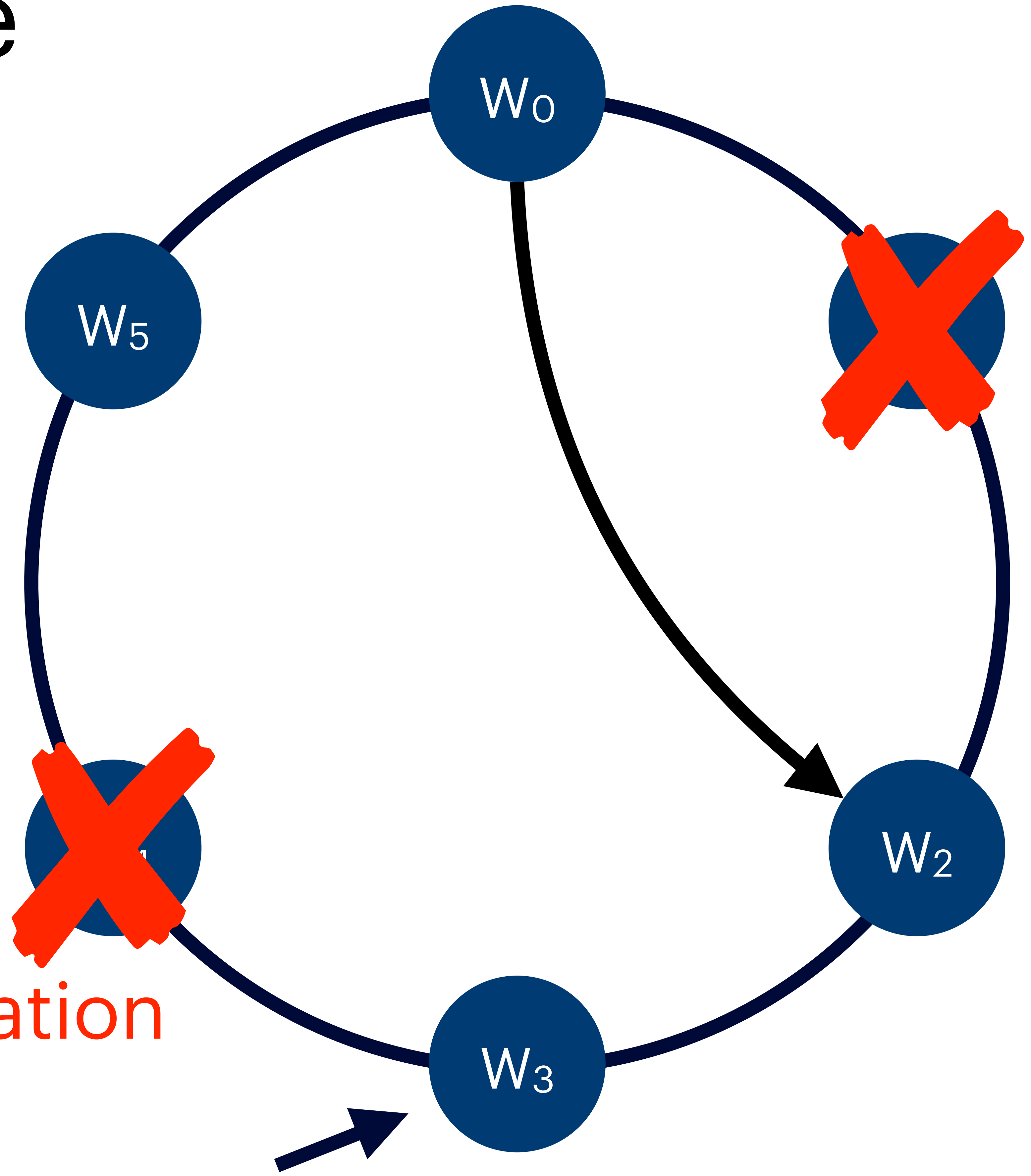


Worker	Membership
0	[0, 2, 3, 4, 5]
2	[0, 2, 3, 4, 5]
3	[0, 2, 3, 4, 5]
4	[0, 1, 2, 3, 4, 5]
5	[0, 1, 2, 3, 4, 5]

Revocation recovery algorithm

Repairing a broken all-reduce ring

Reconcile

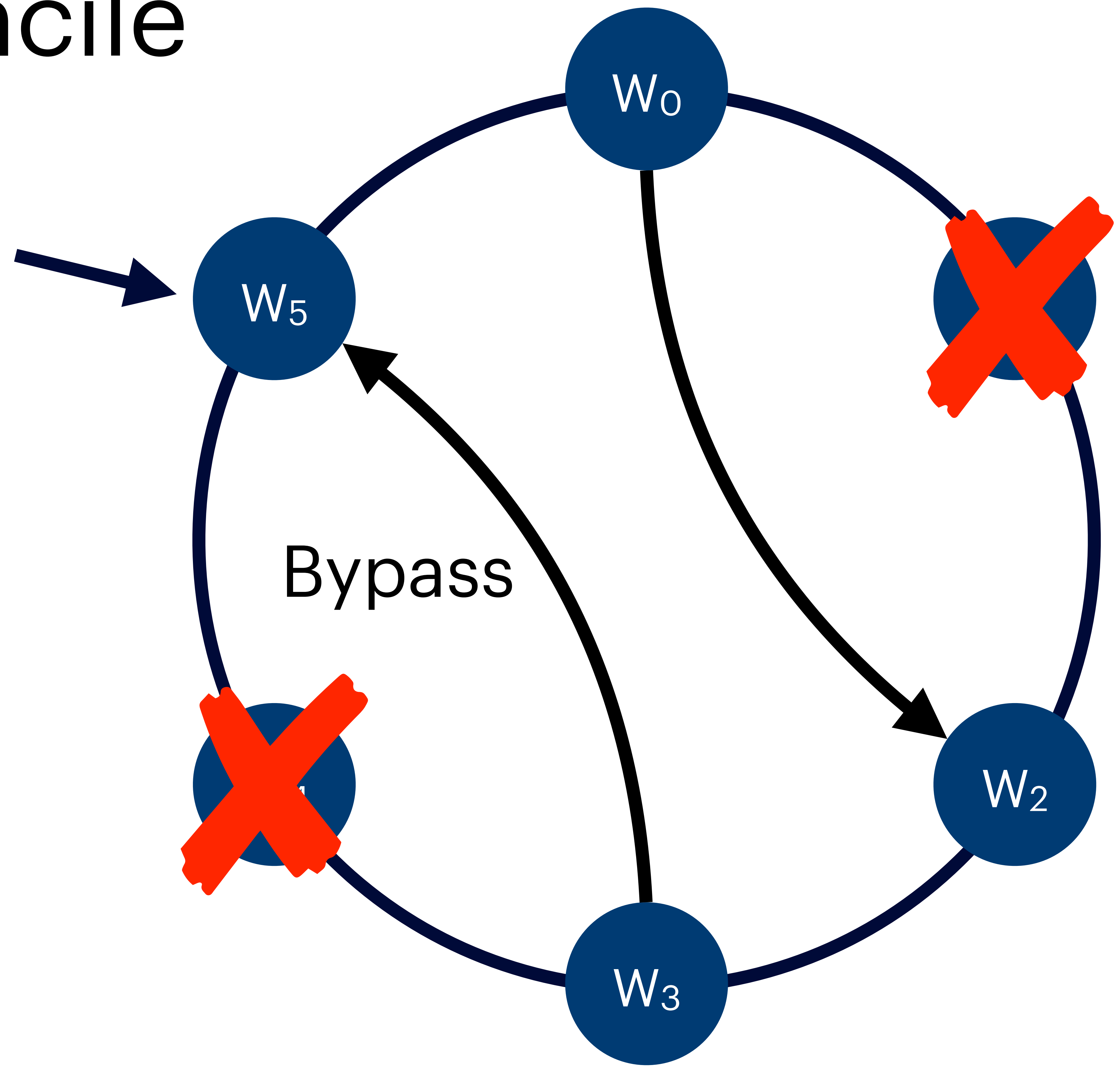


Worker	Membership
0	[0, 2, 3, 4, 5]
2	[0, 2, 3, 4, 5]
3	[0, 2, 3, 4, 5]
4	
5	[0, 1, 2, 3, 4, 5]

Revocation recovery algorithm

Repairing a broken all-reduce ring

Reconcile

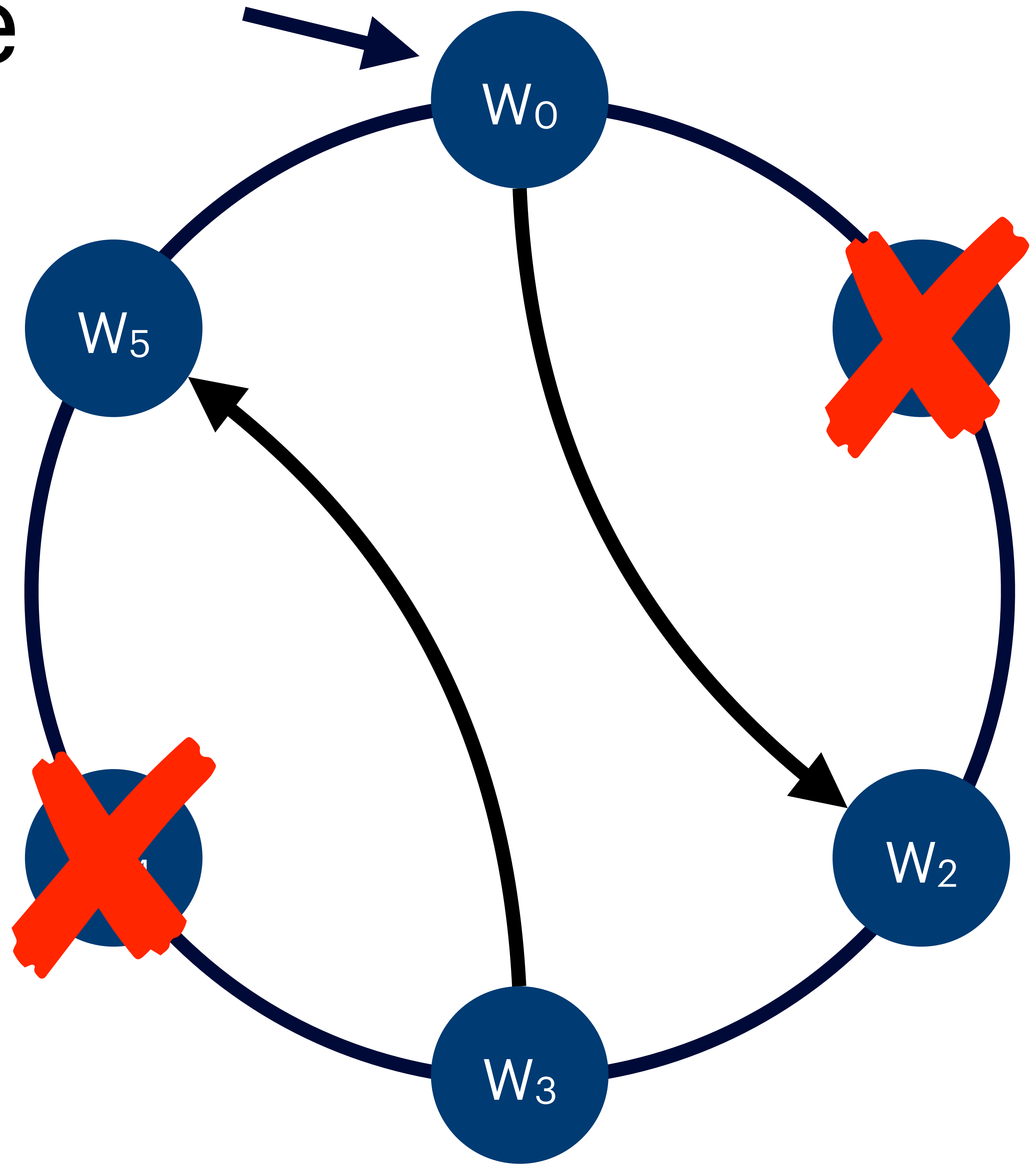


Worker	Membership
0	[0, 2, 3, 4, 5]
2	[0, 2, 3, 4, 5]
3	[0, 2, 3, 5]
4	
5	[0, 2, 3, 5]

Revocation recovery algorithm

Repairing a broken all-reduce ring

Reconcile

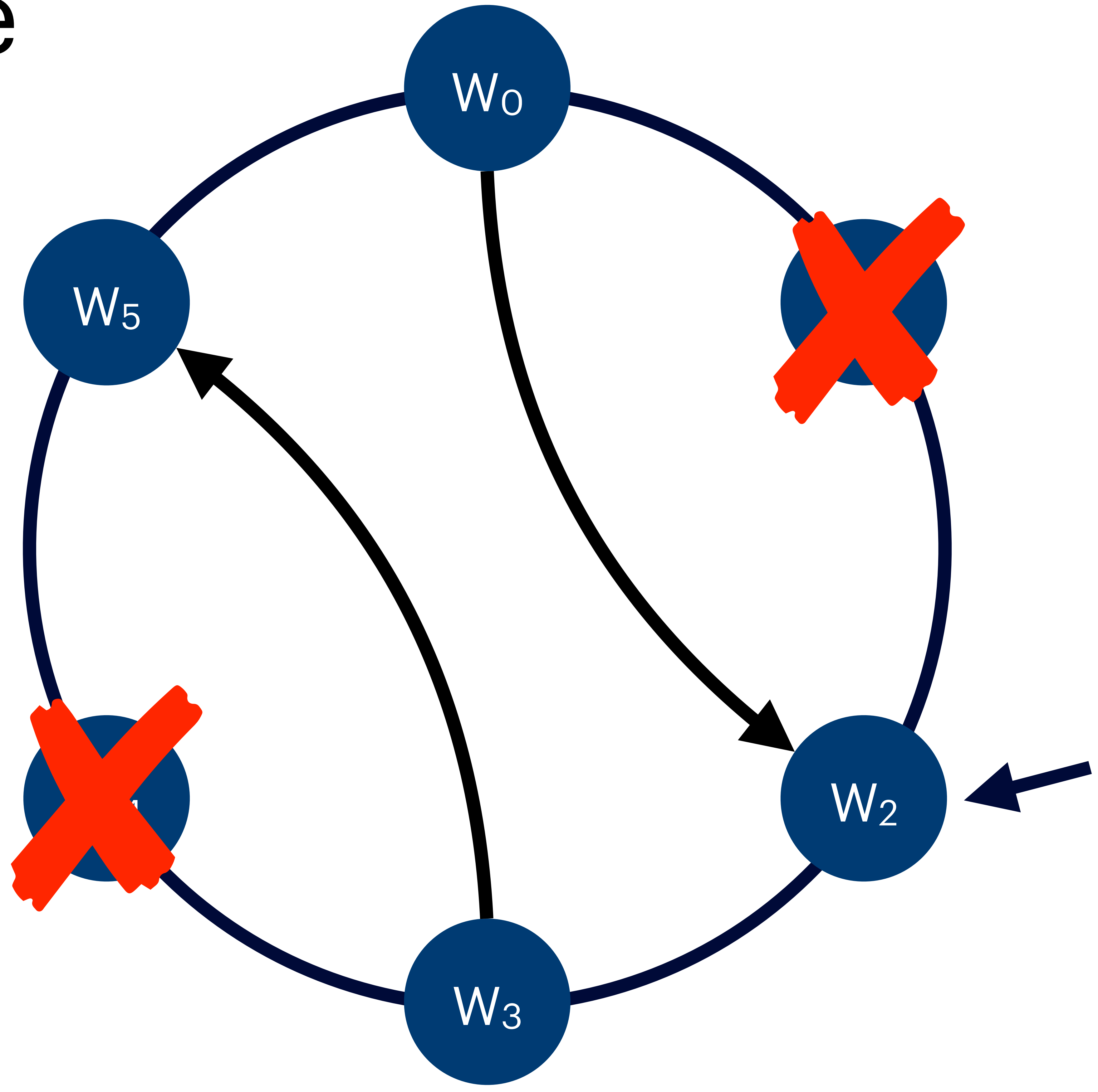


Worker	Membership
0	[0, 2, 3, 5]
2	[0, 2, 3, 4, 5]
3	[0, 2, 3, 5]
4	
5	[0, 2, 3, 5]

Revocation recovery algorithm

Repairing a broken all-reduce ring

Reconcile

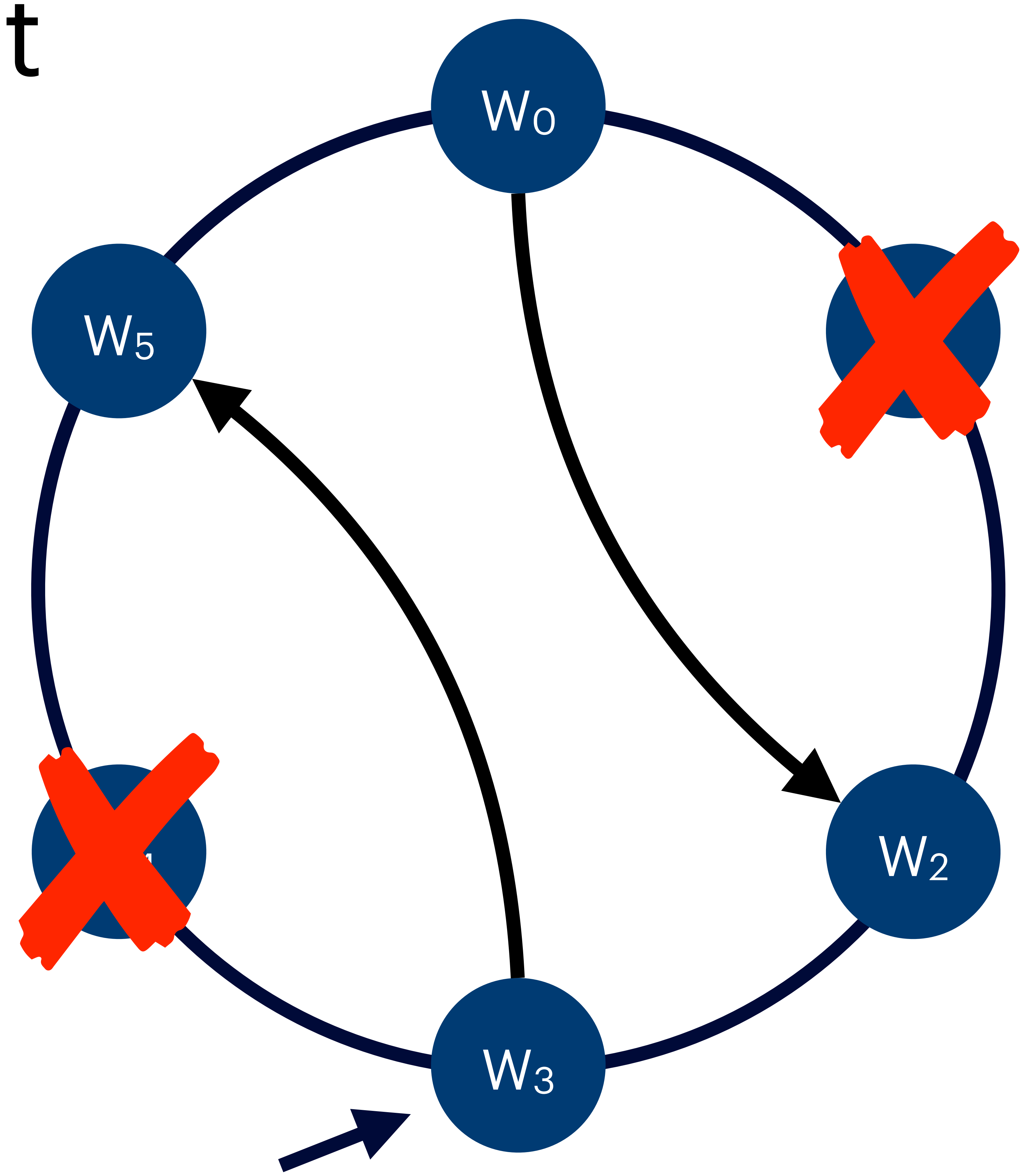


Worker	Membership
0	[0, 2, 3, 5]
2	[0, 2, 3, 5]
3	[0, 2, 3, 5]
4	
5	[0, 2, 3, 5]

Revocation recovery algorithm

Repairing a broken all-reduce ring

Accept

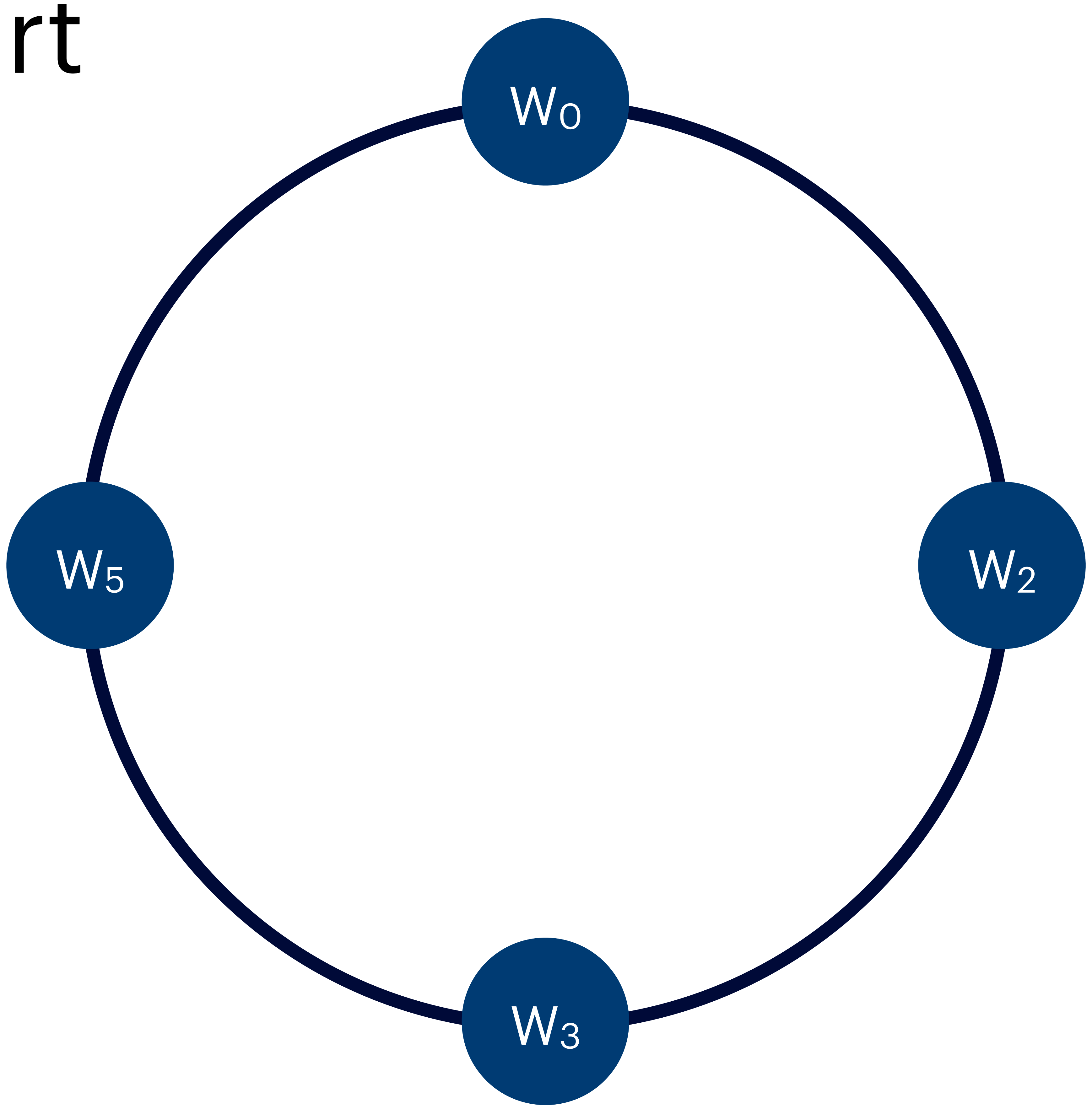


Worker	Membership
0	[0, 2, 3, 5]
2	[0, 2, 3, 5]
3	[0, 2, 3, 5]
4	
5	[0, 2, 3, 5]

Revocation recovery algorithm

Repairing a broken all-reduce ring

Restart



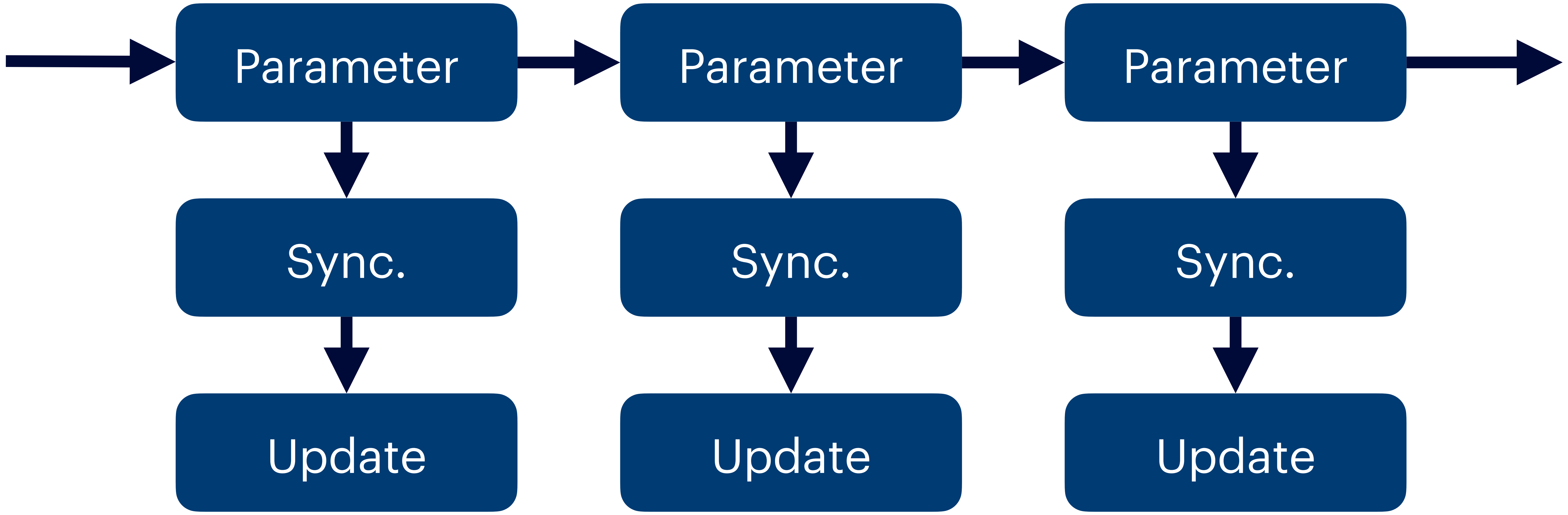
Worker	Membership
0	[0, 2, 3, 5]
2	[0, 2, 3, 5]
3	[0, 2, 3, 5]
5	[0, 2, 3, 5]

Spotnik

Challenges	Solutions
Workers become available or get revoked	**Reuse communication channels** for synchronisation to repair the cluster
Changes can happen at any time	Ensure **atomic model updates** by waiting for all synchronisations to finish first
Cluster sizes are unknown beforehand	**Change the synchronisation strategy** based on the number of workers

Atomic worker state update

- Pipelined synchronisation



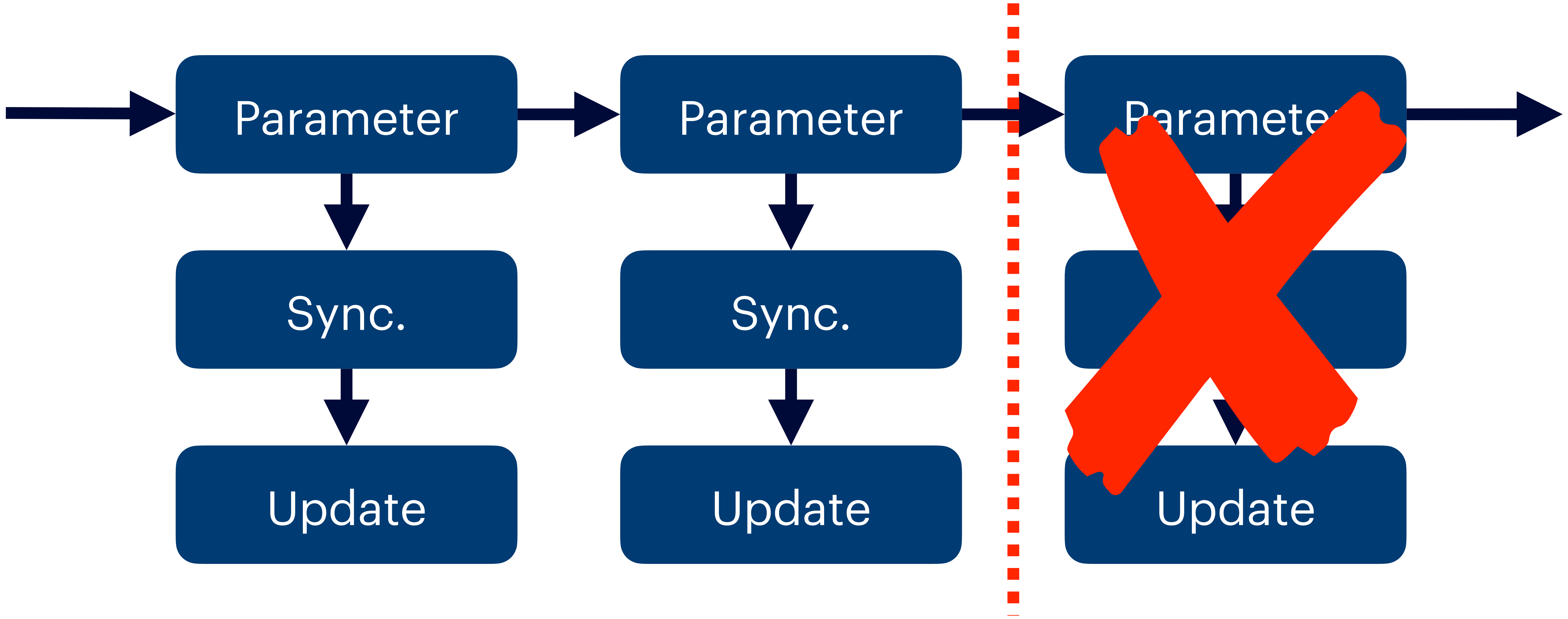
Imperial College London

LSDS Large-Scale Distributed Systems Group

Atomic worker state update

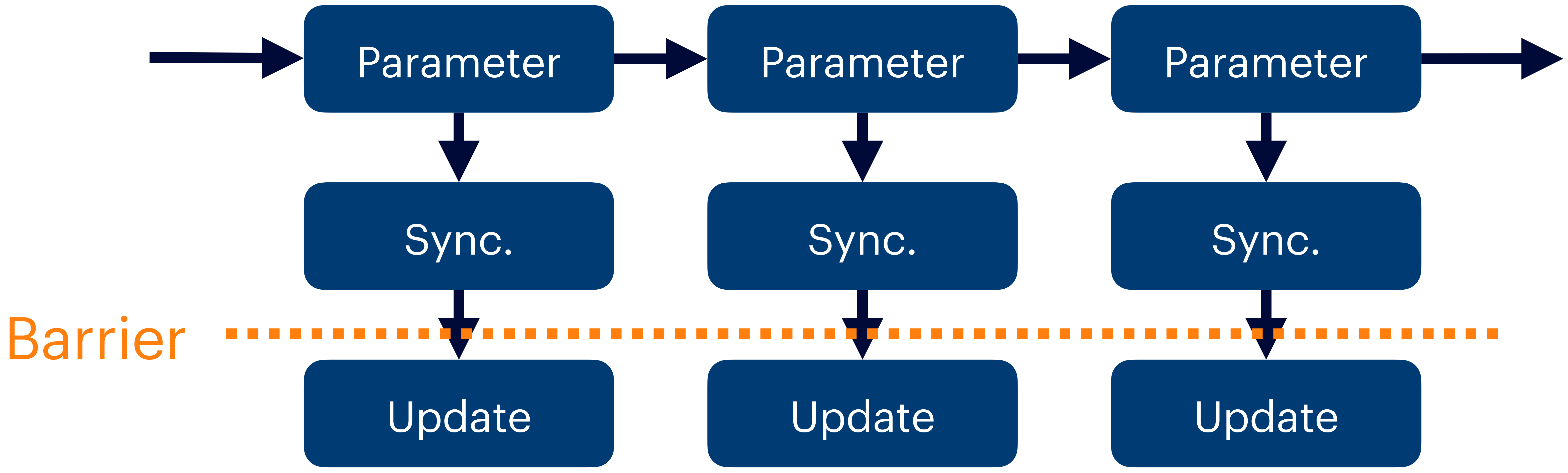
- Pipelined synchronisation
- Revocations can happen meanwhile

→ Partial update leads to inconsistency



Atomic worker state update

- Atomicity: Wait for all synchronisation communications to finish
- Discard updates if communication fails



Imperial College London

LSDS Large-Scale Distributed Systems Group

Spotnik

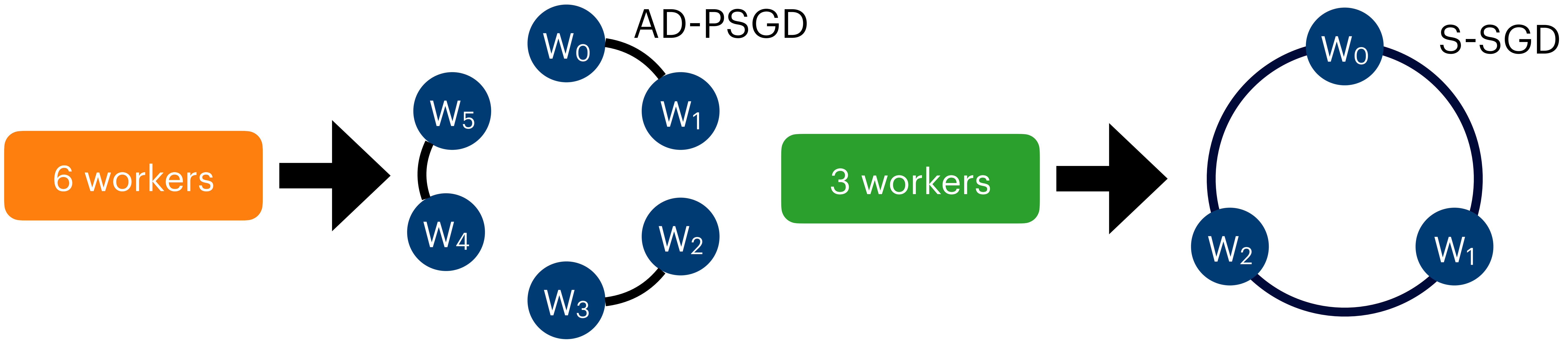
Challenges	Solutions
Workers become available or get revoked	**Reuse communication channels** for synchronisation to repair the cluster
Changes can happen at any time	Ensure **atomic model updates** by waiting for all synchronisations to finish first
Cluster sizes are unknown beforehand	**Change the synchronisation strategy** based on the number of workers

Adaptive synchronisation strategies

- Support a range of synchronisation primitives
  - collective and point-to-point synchronisation
- Monitor a metric
  - Number of workers

Adaptive synchronisation strategies

- Support a range of synchronisation primitives
  - collective and point-to-point synchronisation
- Monitor a metric
  - Number of workers
- Define a policy in the beginning
  - When to choose which sync strategy



Evaluation

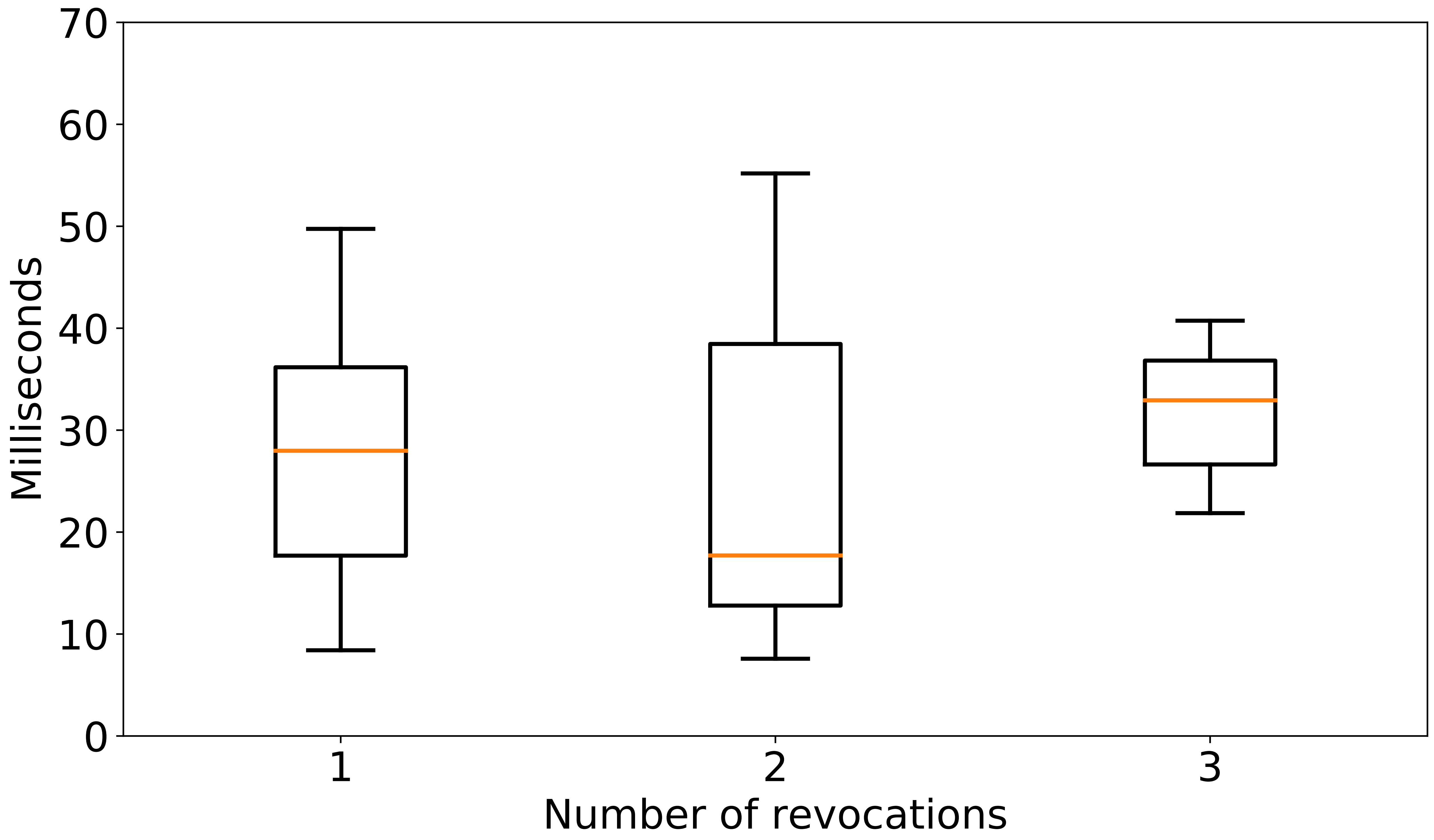
How does the recovery latency change with increasing number of revocations?

Cluster
- 16 workers

Hardware
- Azure NC6 VMs
  - Nvidia K80

Software
- KungFu 0.2.1
- Tensorflow 1.15

ML
- ResNet50
- ImageNet



No significant increase of recovery latency if the number of revocation increases

Imperial College London

LSDS Large-Scale Distributed Systems Group

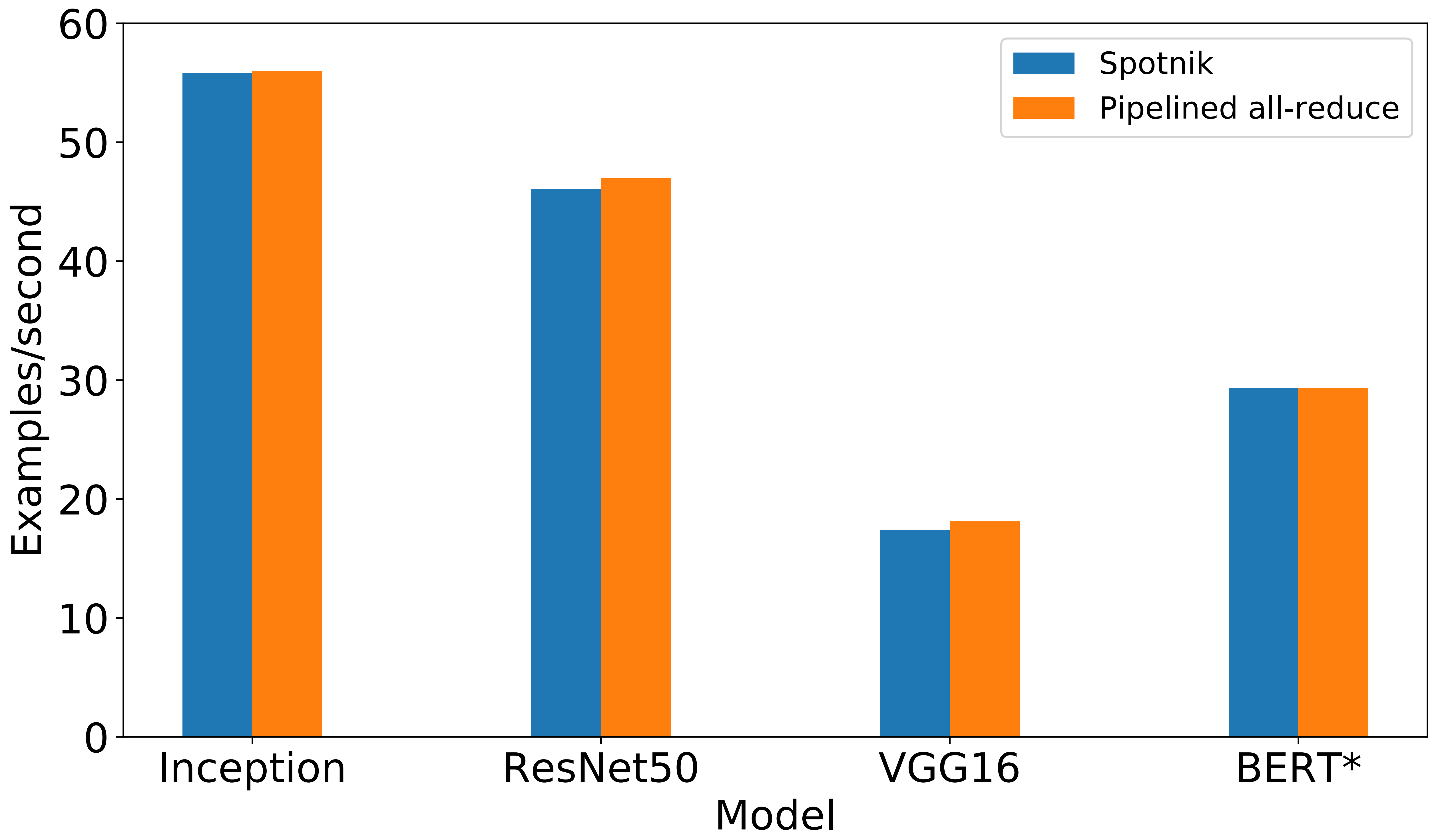
Evaluation

How much does the training slow down if we use atomic worker state updates?

Cluster
- 32 workers

Hardware
- Azure NC6 VMs
  - Nvidia K80

Software
- KungFu 0.2.1
- Tensorflow 1.15



*different Setup

Cluster
- 16 workers

Hardware
- Huawei ModelArts
  - Nvidia V100
  - InfiniBand

Software
- KungFu 0.2.1
- Tensorflow 1.12

Throughput decrease is small

Evaluation

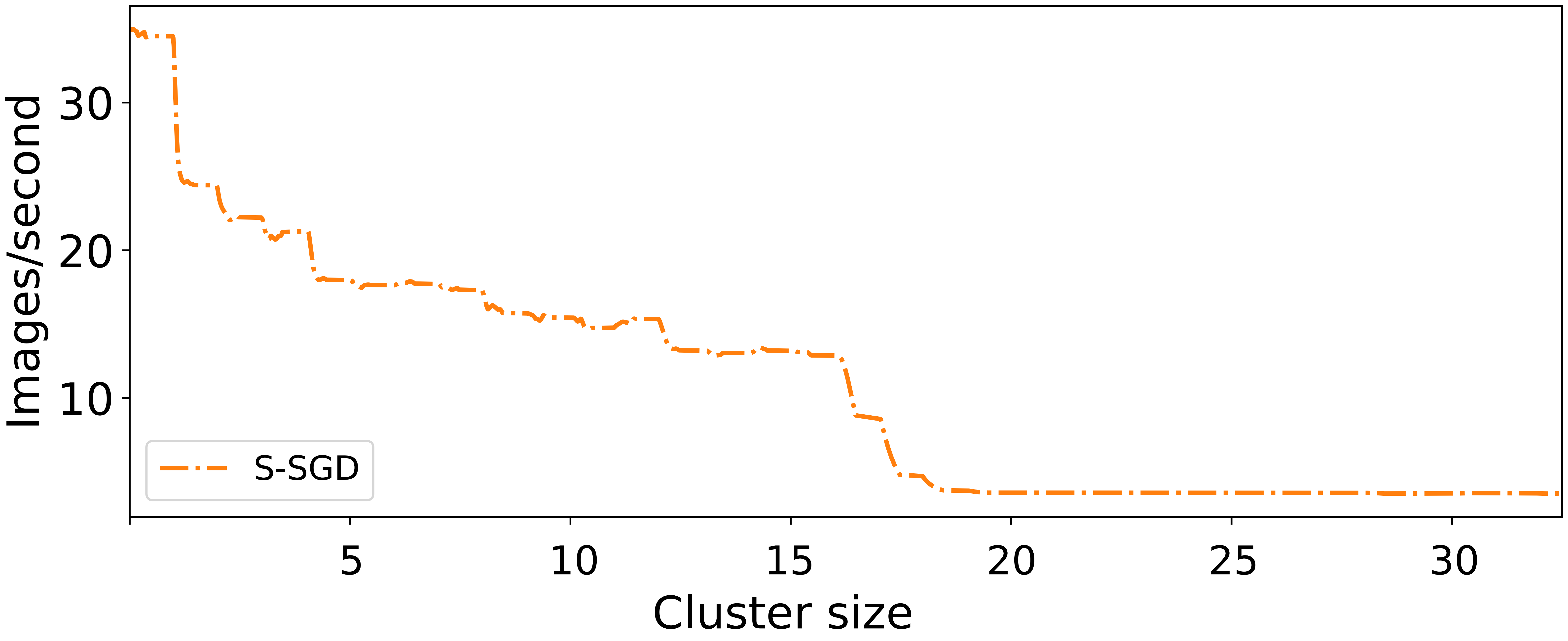
How does the throughput change, if the cluster changes?

Cluster
- up to 32 workers

Hardware
- Azure NC6 VMs
    - Nvidia K80

Software
- KungFu 0.2.1
- Tensorflow 1.15

ML
- ResNet50
- ImageNet



Imperial College London

LSDS Large-Scale Distributed Systems Group

Evaluation

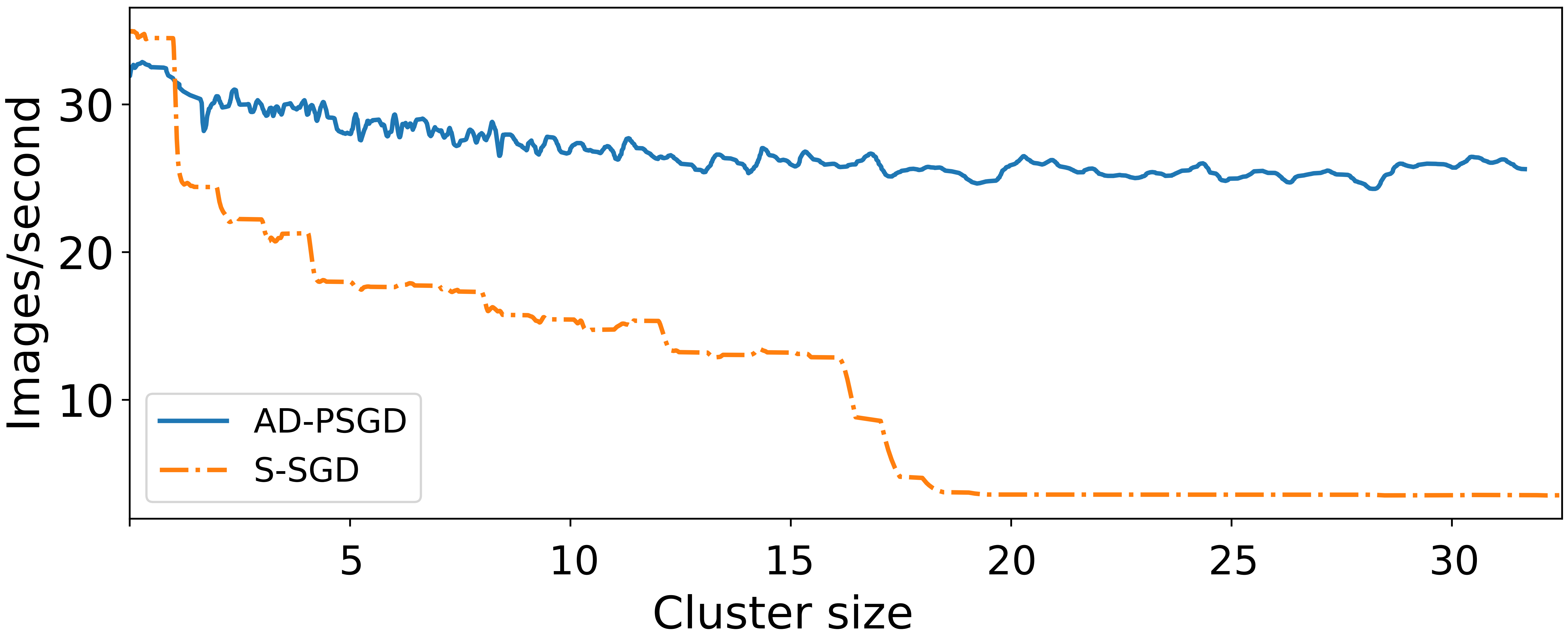
How does the throughput change, if the cluster changes?

Cluster
- up to 32 workers

Hardware
- Azure NC6 VMs
  - Nvidia K80

Software
- KungFu 0.2.1
- Tensorflow 1.15

ML
- ResNet50
- ImageNet



Evaluation

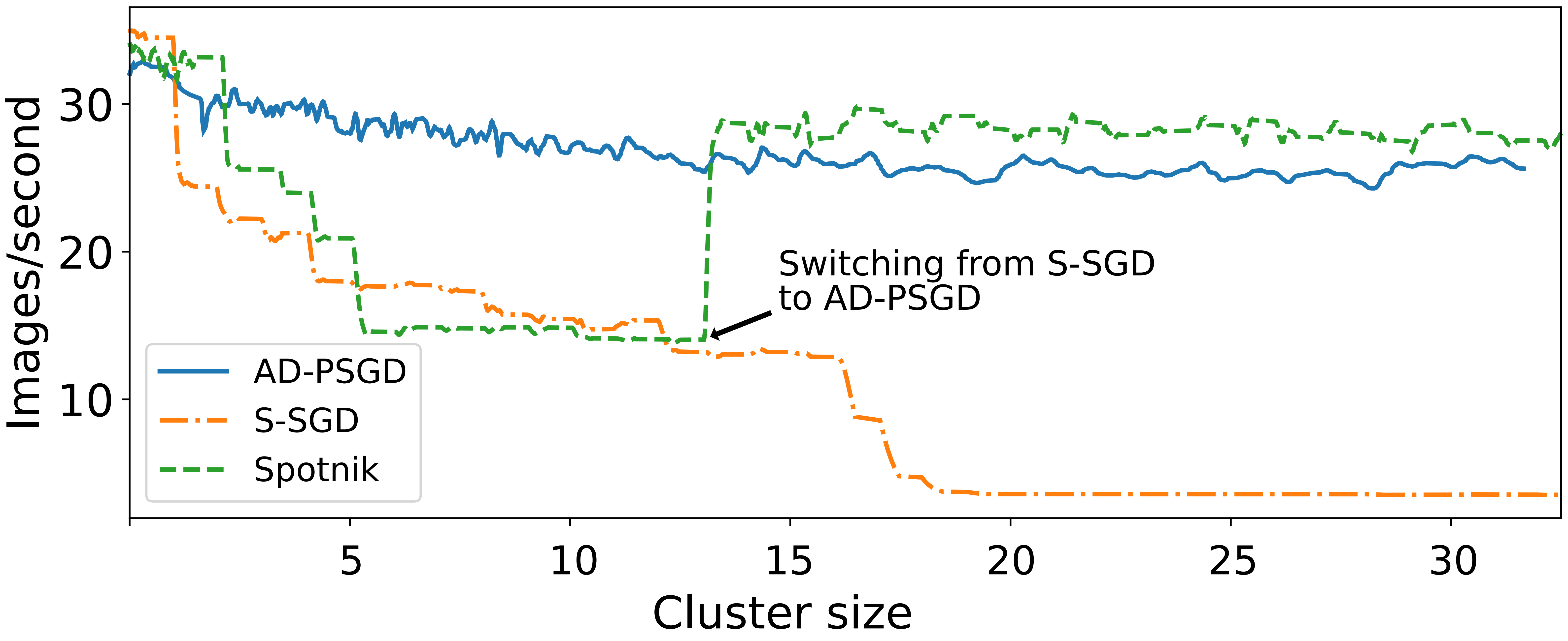
How does the throughput change, if the cluster changes?

Cluster
- up to 32 workers

Hardware
- Azure NC6 VMs
  - Nvidia K80

Software
- KungFu 0.2.1
- Tensorflow 1.15

ML
- ResNet50
- ImageNet



Changing clusters need adaptation

Conclusion

- Transient cloud resources offer potential to save money for ML training
- No system that runs exclusively on transient resources and has low overhead

Conclusion

- Transient cloud resources offer potential to save money for ML training
- No system that runs exclusively on transient resources and has low overhead

Spotnik

- Repair cluster with low overhead
- Ensure consistent model updates
- Adapt to changes of the cluster

Conclusion

- Transient cloud resources offer potential to save money for ML training
- No system that runs exclusively on transient resources and has low overhead

Spotnik

- Repair cluster with low overhead
- Ensure consistent model updates
- Adapt to changes of the cluster

KungFu github.com/lsds/KungFu

Conclusion

- Transient cloud resources offer potential to save money for ML training
- No system that runs exclusively on transient resources and has low overhead

Spotnik

- Repair cluster with low overhead
- Ensure consistent model updates
- Adapt to changes of the cluster

KungFu github.com/lsds/KungFu

Website lsds.doc.ic.ac.uk | E-Mail marcel.wagenlander19@imperial.ac.uk
Twitter @marwage